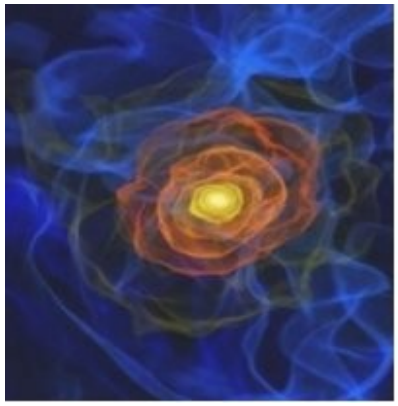
Memory Efficient GPU-Based Ray Casting for Unstructured Volume Rendering



IEEE/EG International Symposium on Volume Graphics

10 - 11 August, 2008 (co-located with SIGGRAPH 2008)

Los Angeles, CA, USA

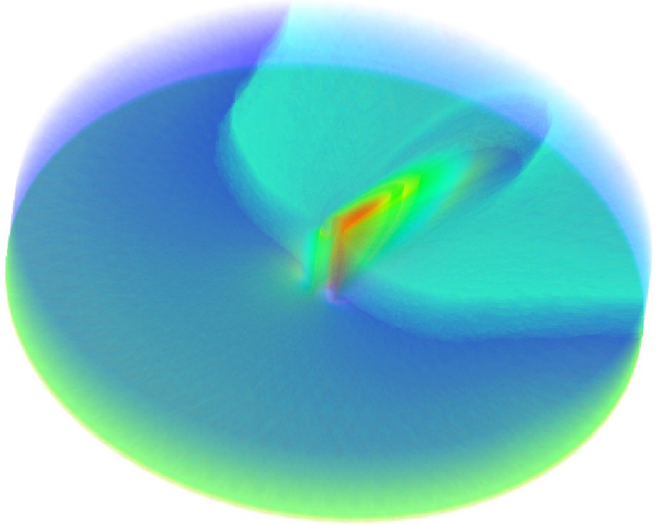
André Maximo
Saulo Ribeiro
Cristiana Bentes
Antonio Oliveira
Ricardo Farias

UFRJ COPPE

LABORATÓRIO DE COMPUTAÇÃO GRÁFICA
PESC / COPPE / UFRJ

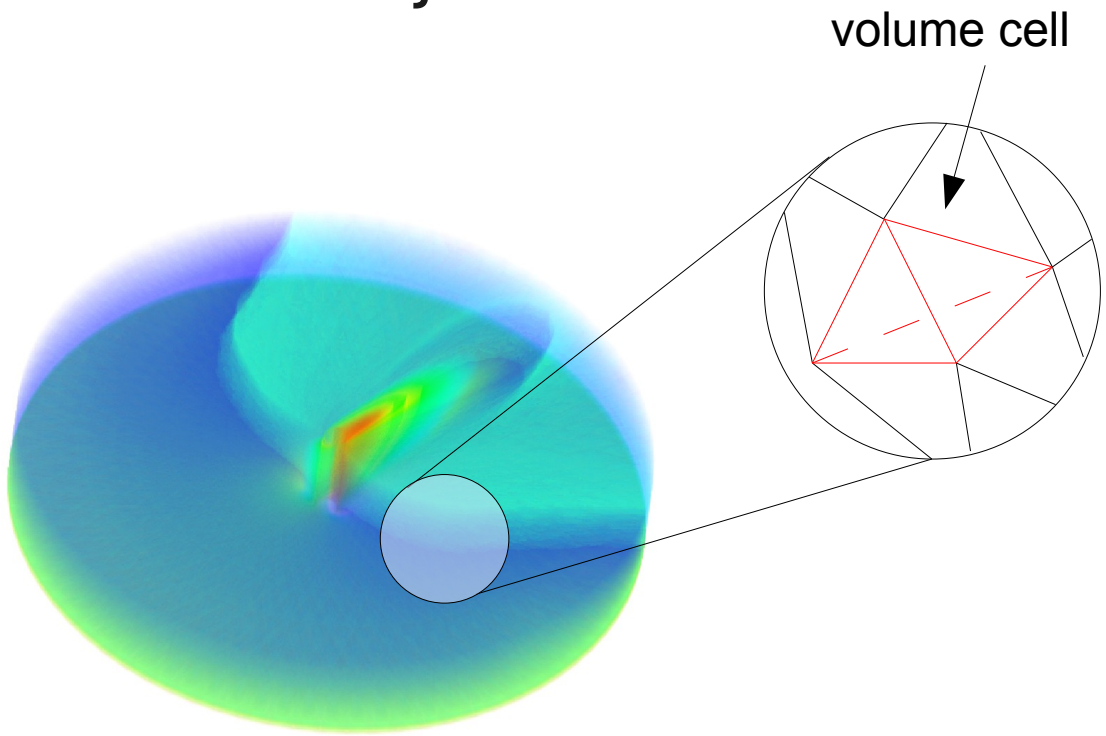
Introduction

- Direct Volume Rendering approaches
  - Cell Projection



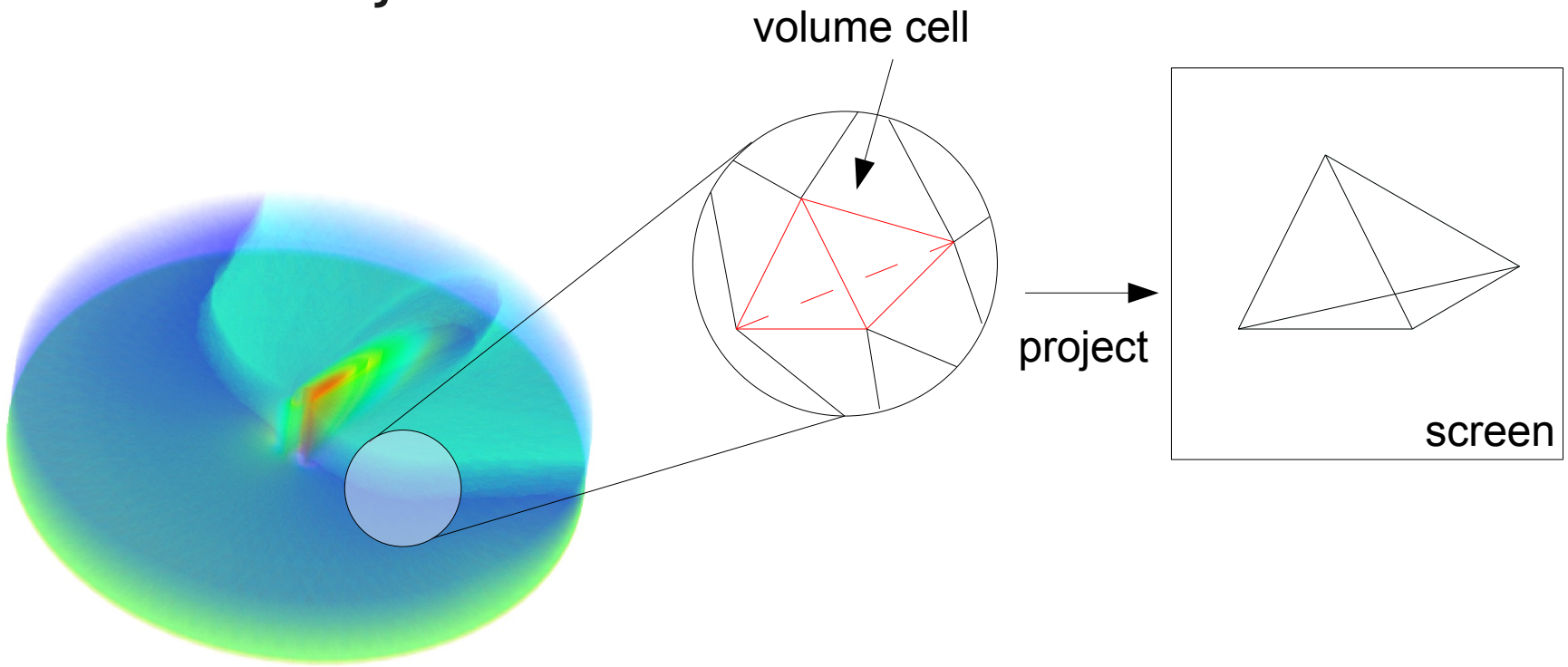
Introduction

- Direct Volume Rendering approaches
  - Cell Projection



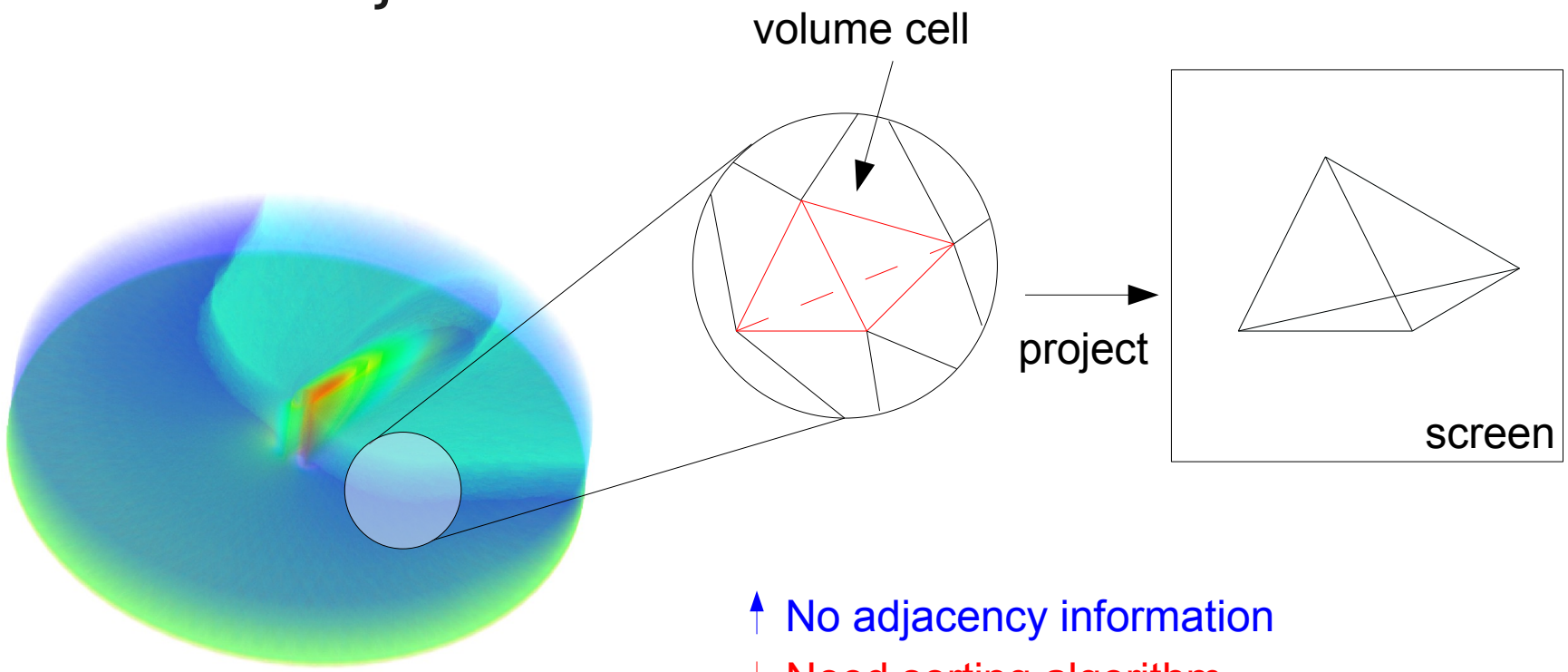
Introduction

- Direct Volume Rendering approaches
  - Cell Projection



Introduction

- Direct Volume Rendering approaches
  - Cell Projection

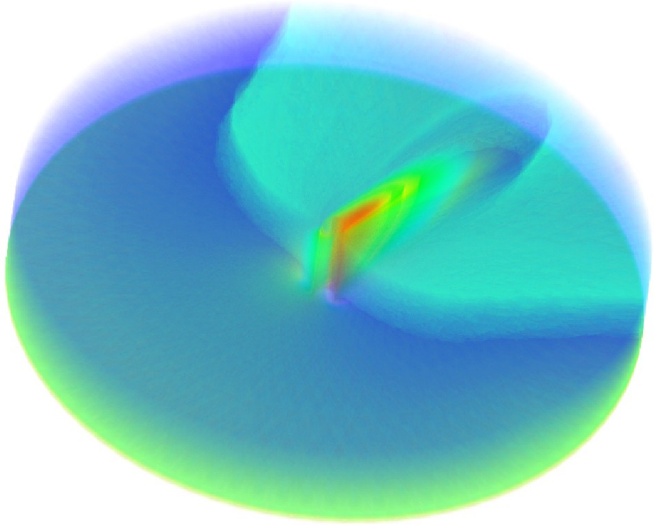


No adjacency information

Need sorting algorithm

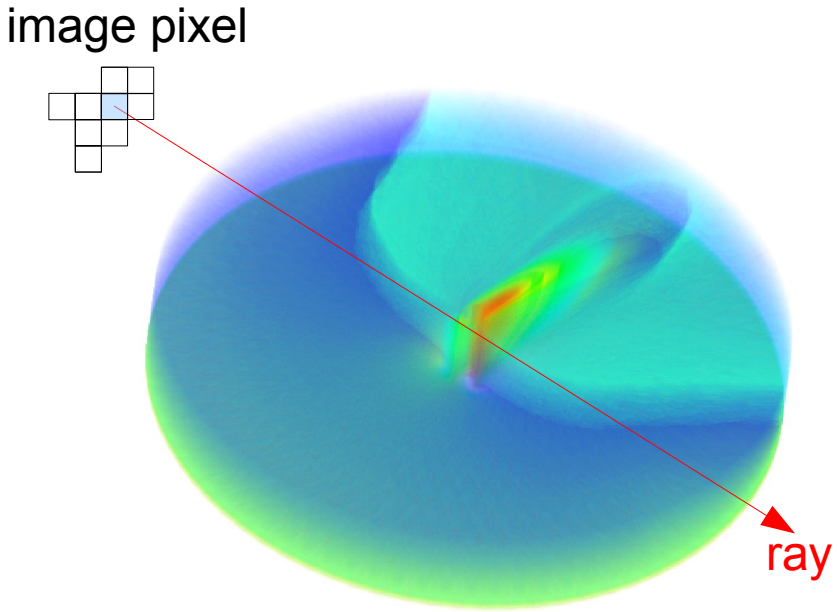
Introduction

- Direct Volume Rendering approaches
  - Ray Casting



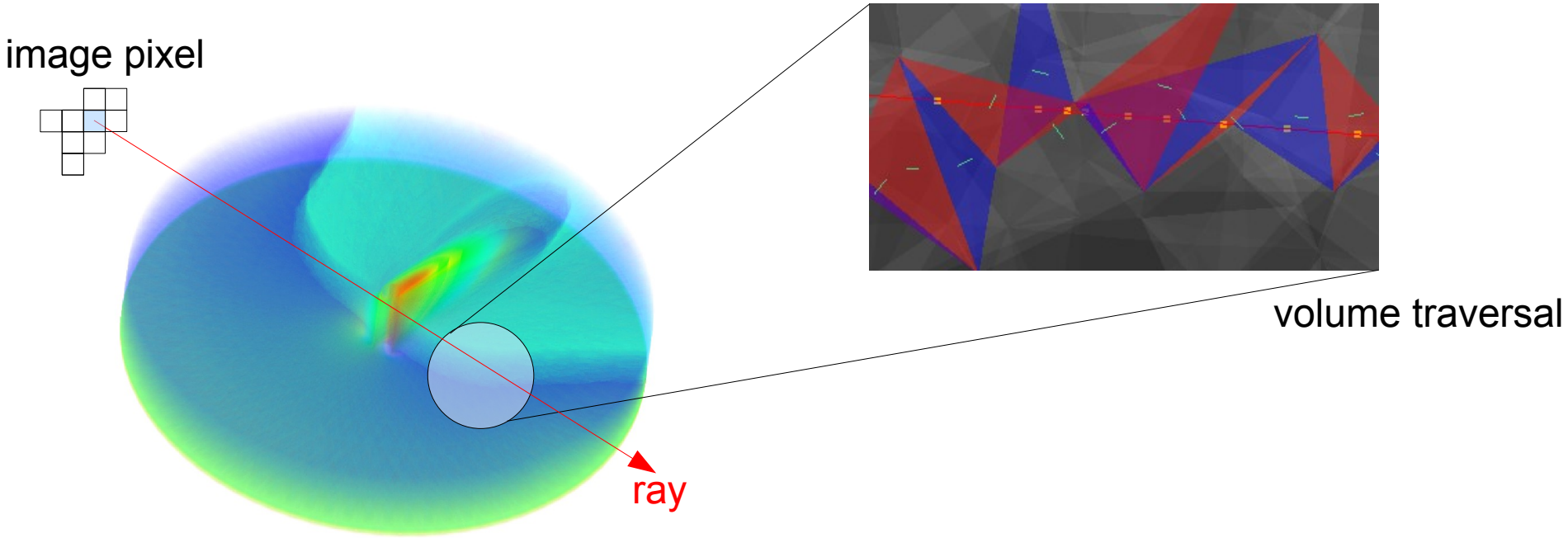
Introduction

- Direct Volume Rendering approaches
  - Ray Casting



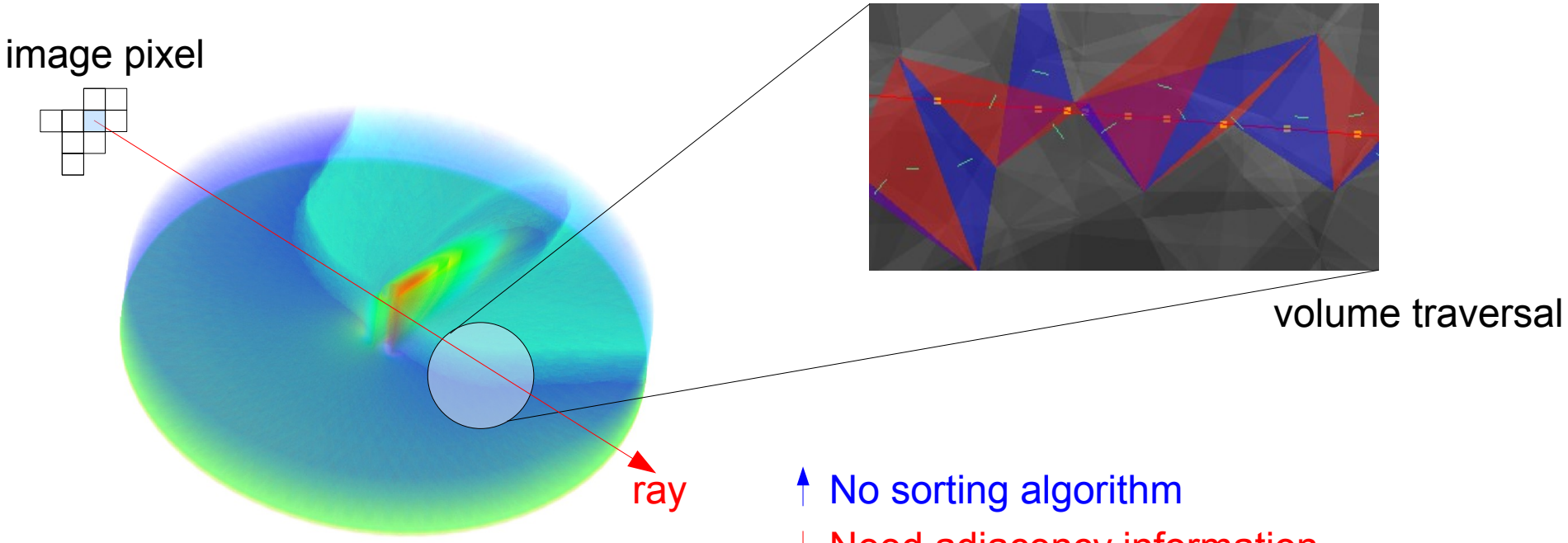
Introduction

- Direct Volume Rendering approaches
  - Ray Casting



Introduction

- Direct Volume Rendering approaches
  - Ray Casting



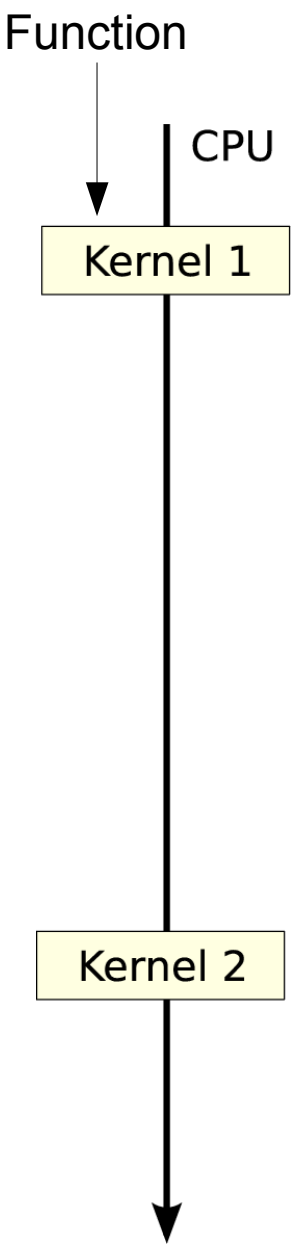
↑ No sorting algorithm

↓ Need adjacency information

Objectives

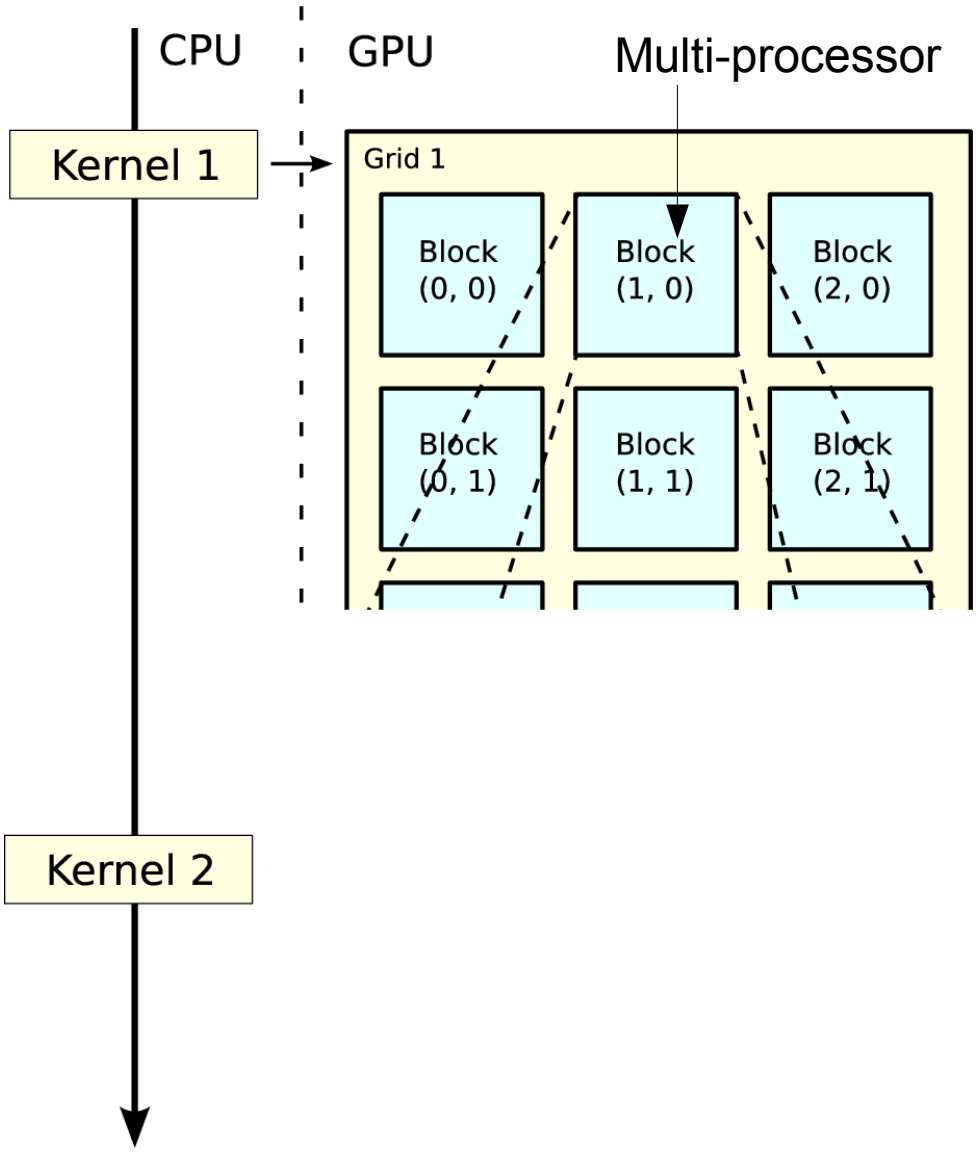
- Volume Rendering – Ray Casting
- Interactive performance – GPU
- Memory usage – Large Datasets
- Base algorithm – VF-Ray

A GPU-Based approach



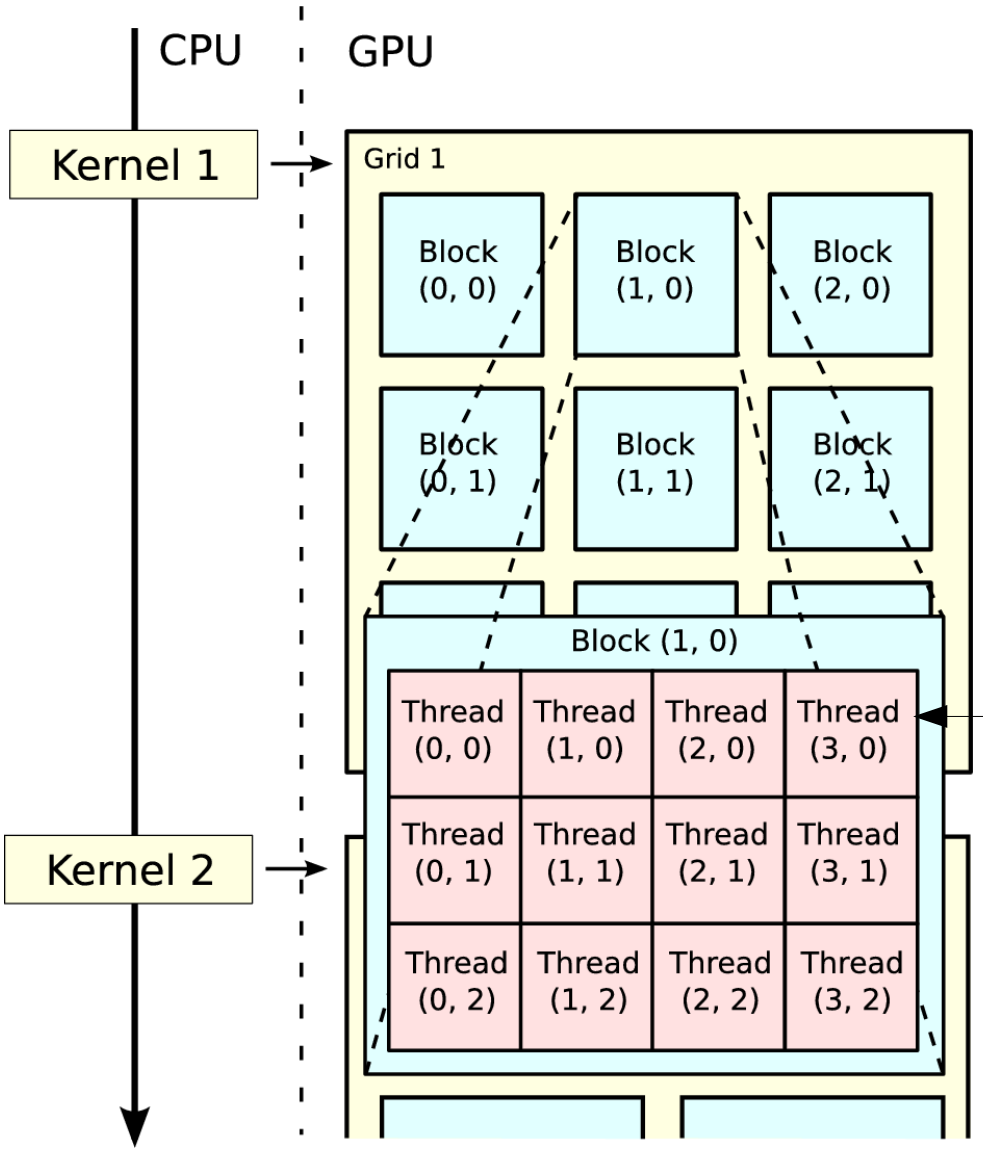
- CUDA – Compute Unified Device Architecture

A GPU-Based approach



- CUDA – Compute Unified Device Architecture

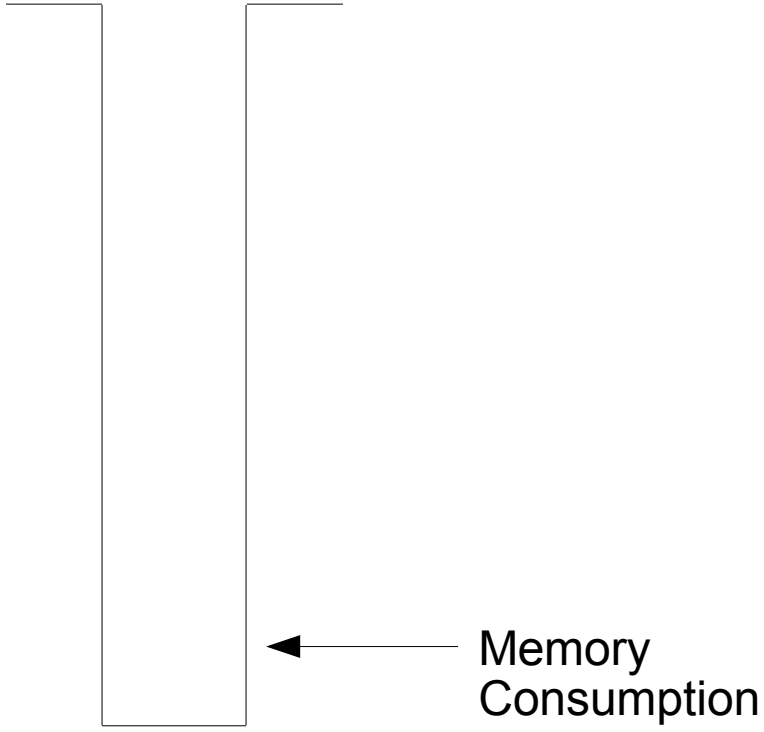
A GPU-Based approach



- CUDA – Compute Unified Device Architecture

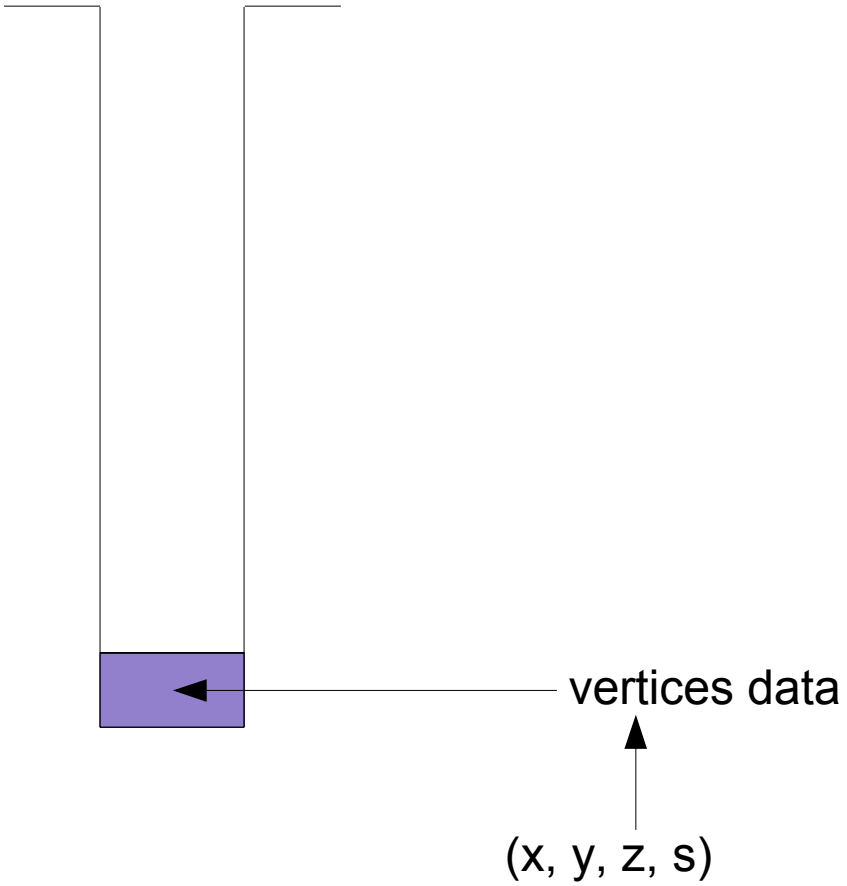
Concerning Memory

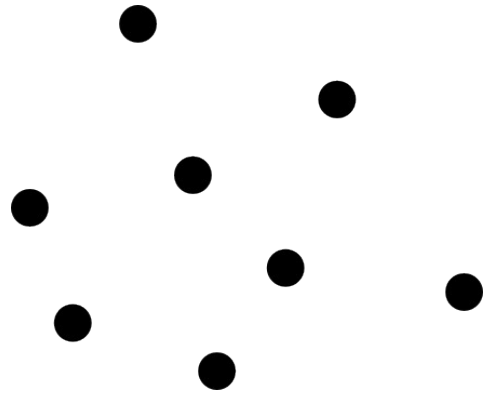
- Volume Rendering “Memory Well”



Concerning Memory

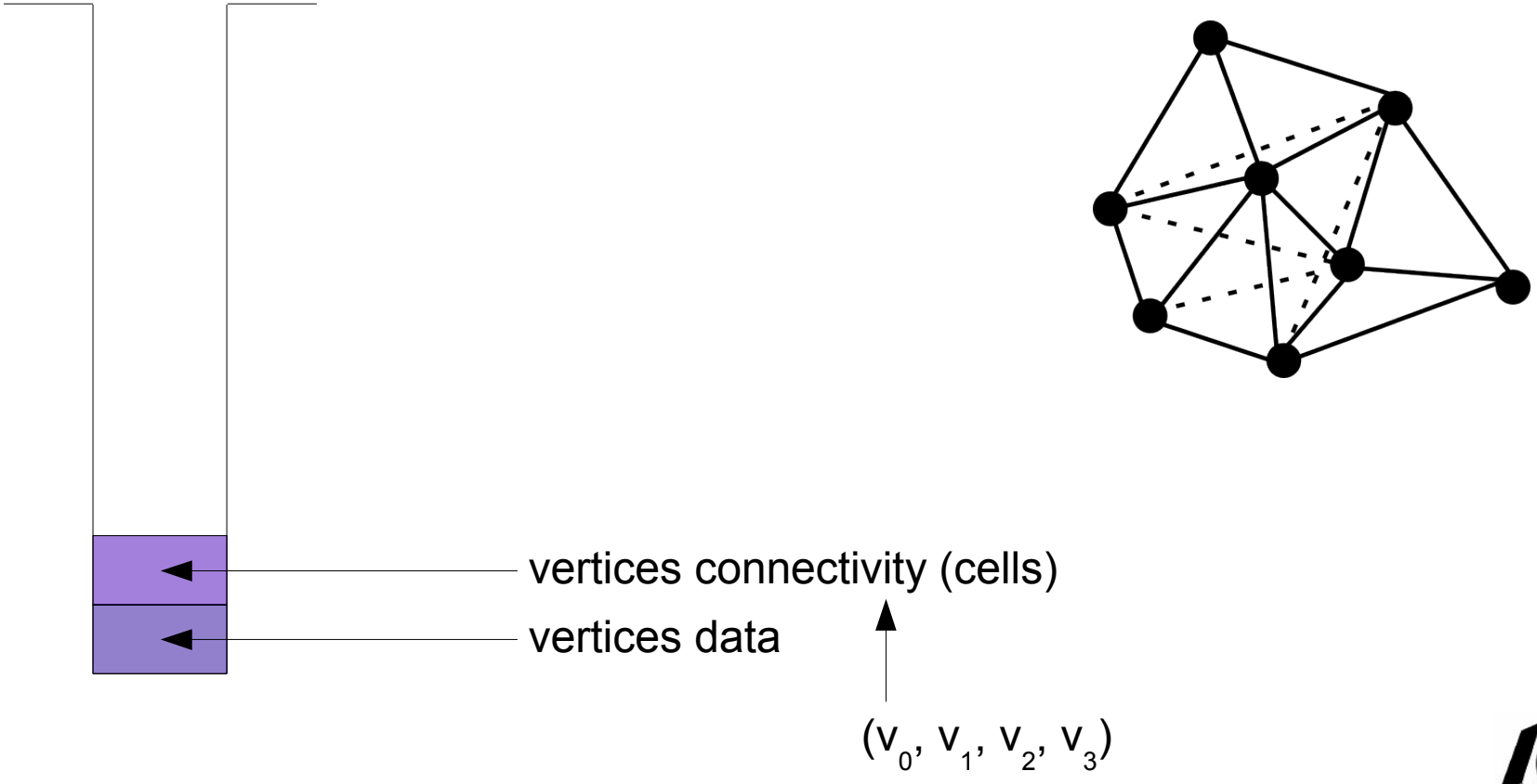
- Volume Rendering “Memory Well”





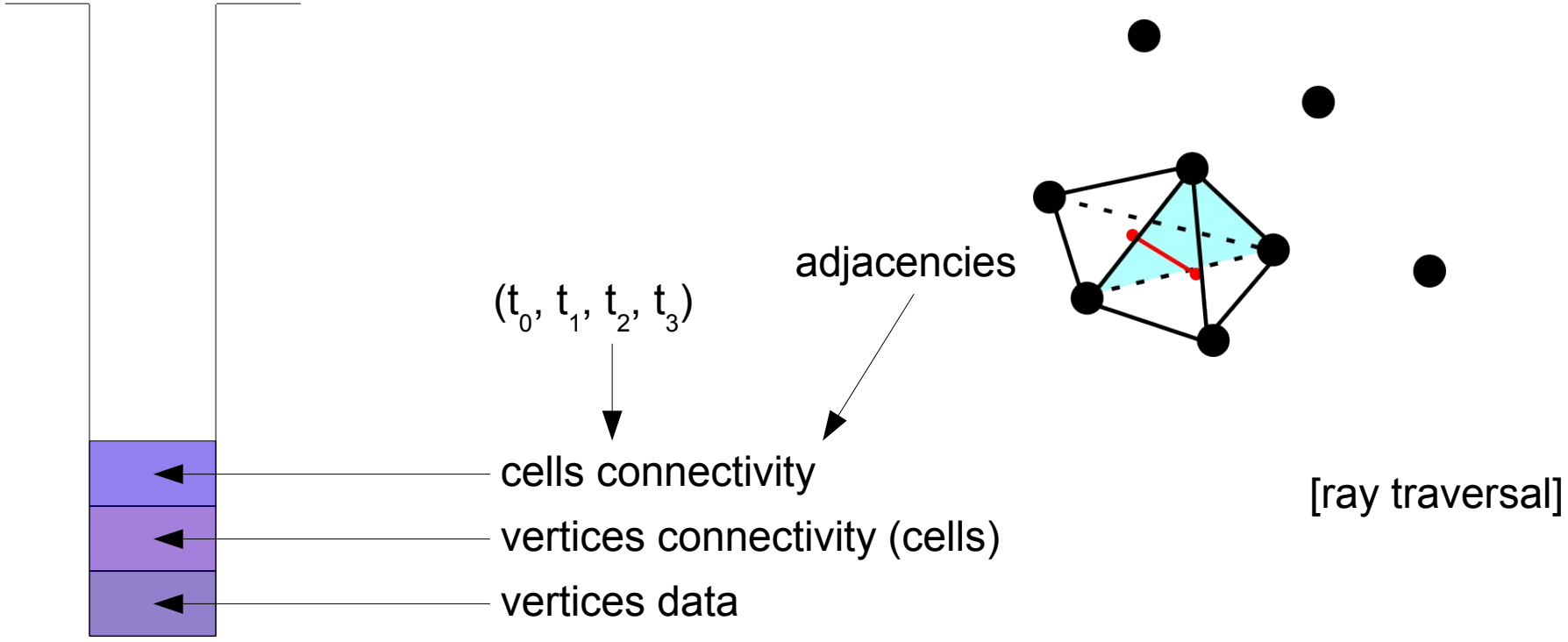
Concerning Memory

- Volume Rendering “Memory Well”



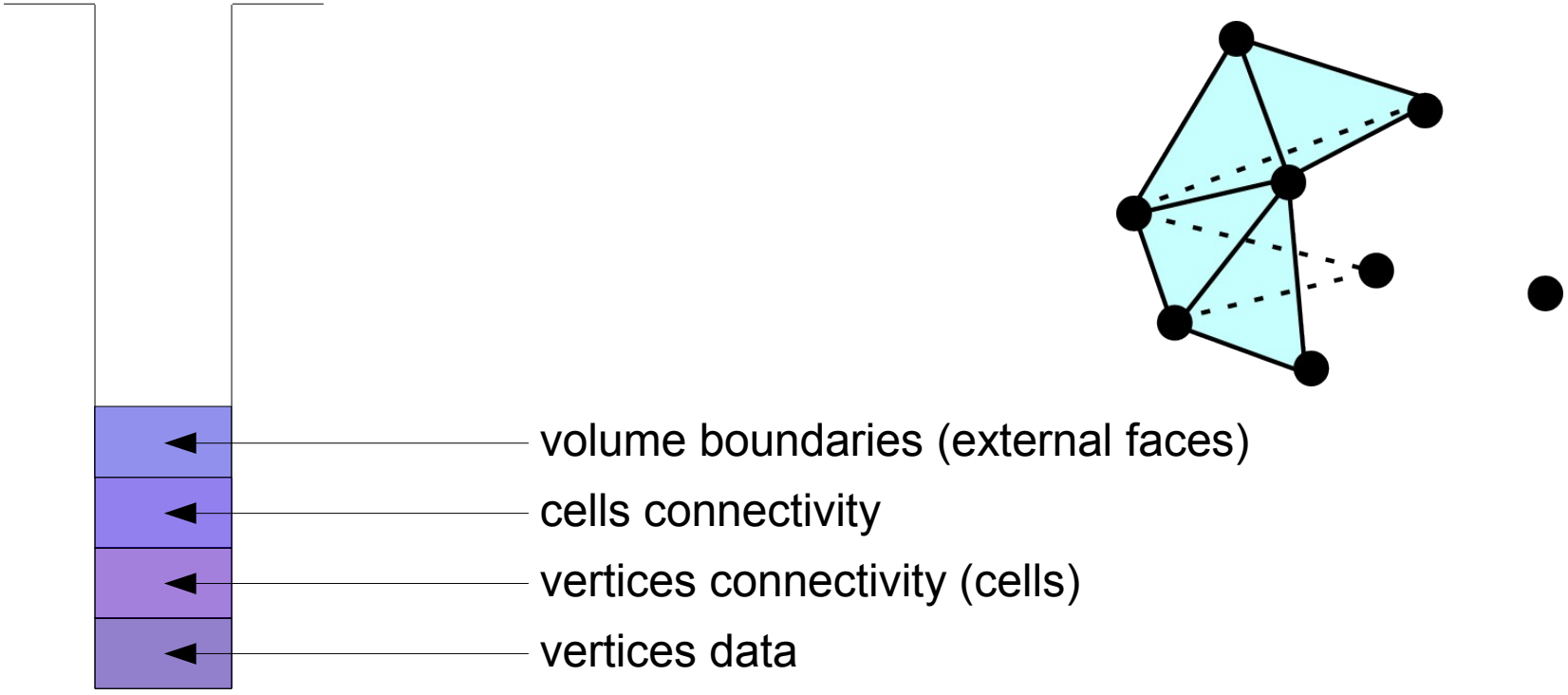
Concerning Memory

- Volume Rendering “Memory Well”



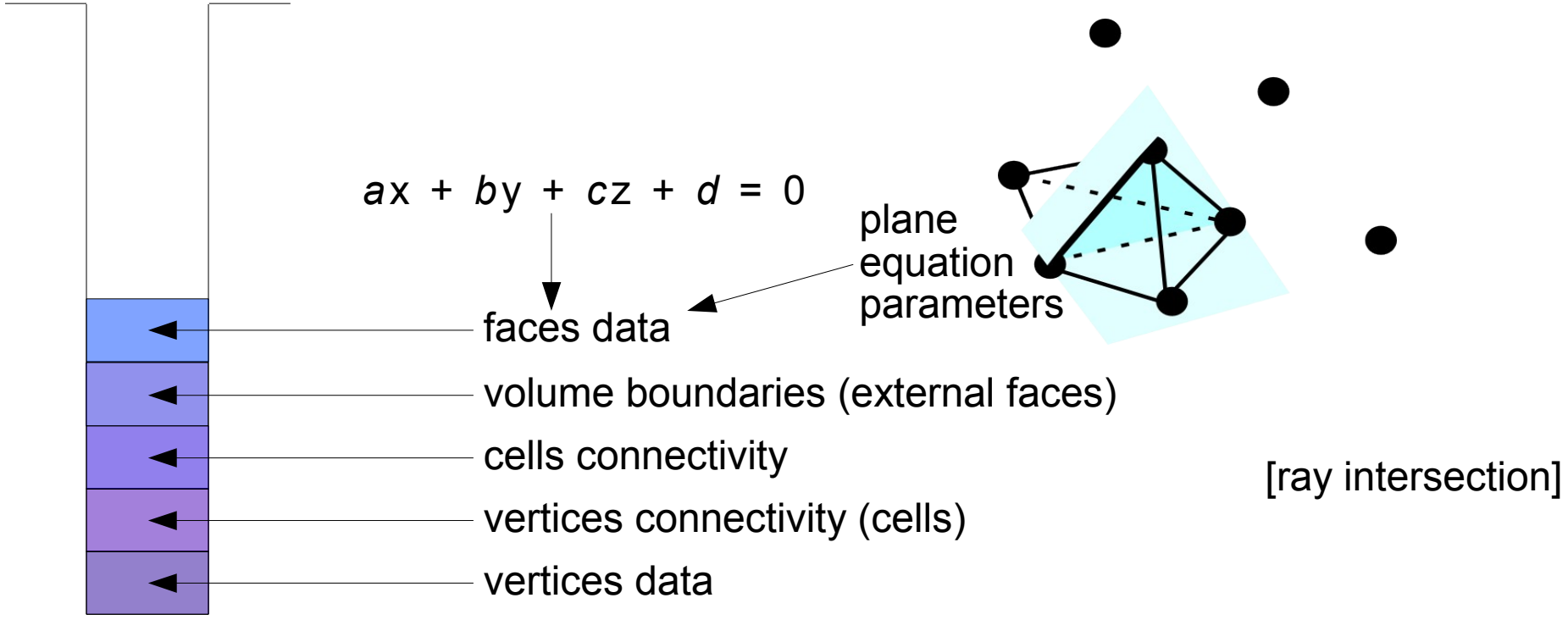
Concerning Memory

- Volume Rendering “Memory Well”



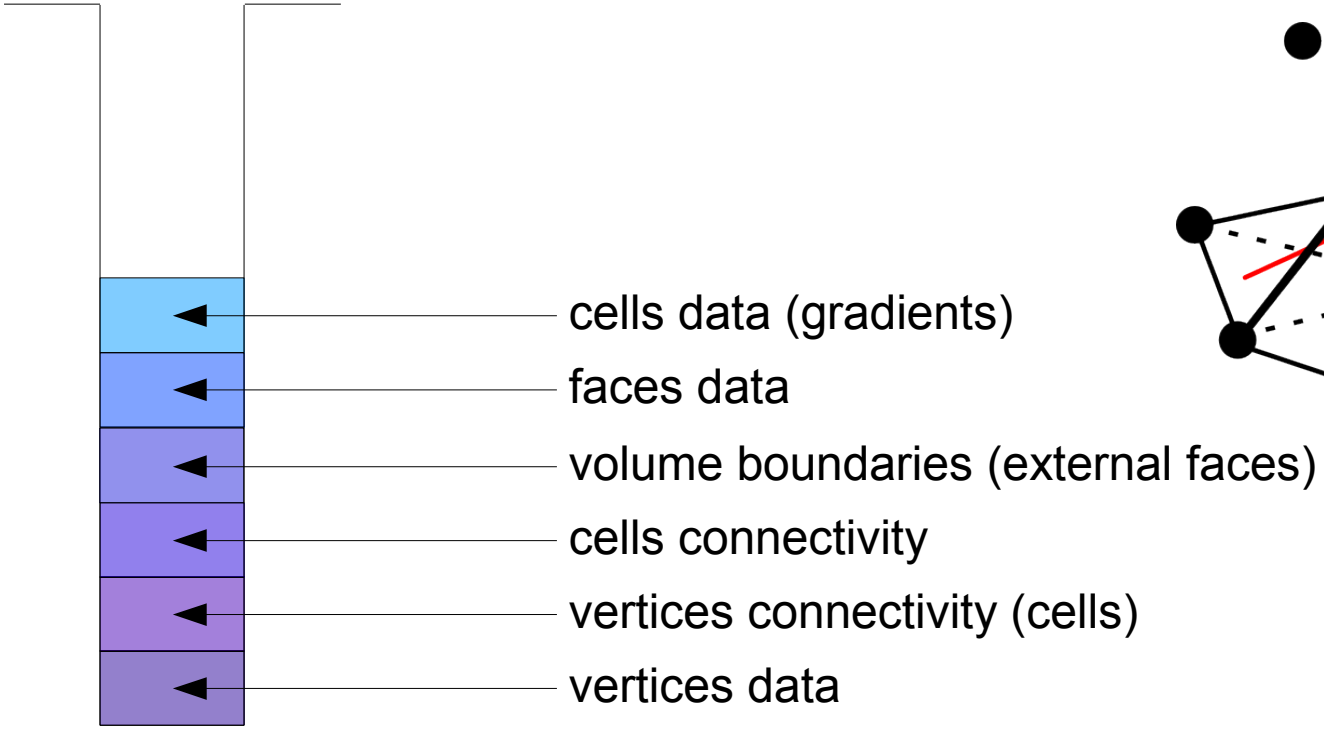
Concerning Memory

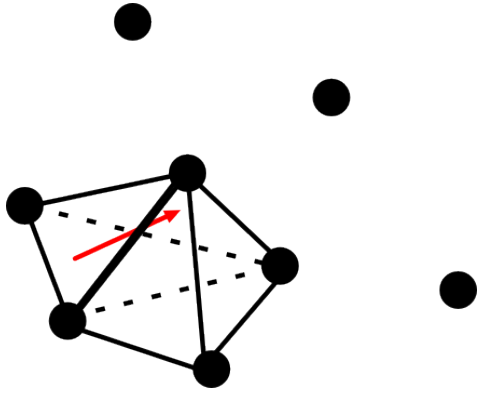
- Volume Rendering “Memory Well”



Concerning Memory

- Volume Rendering “Memory Well”

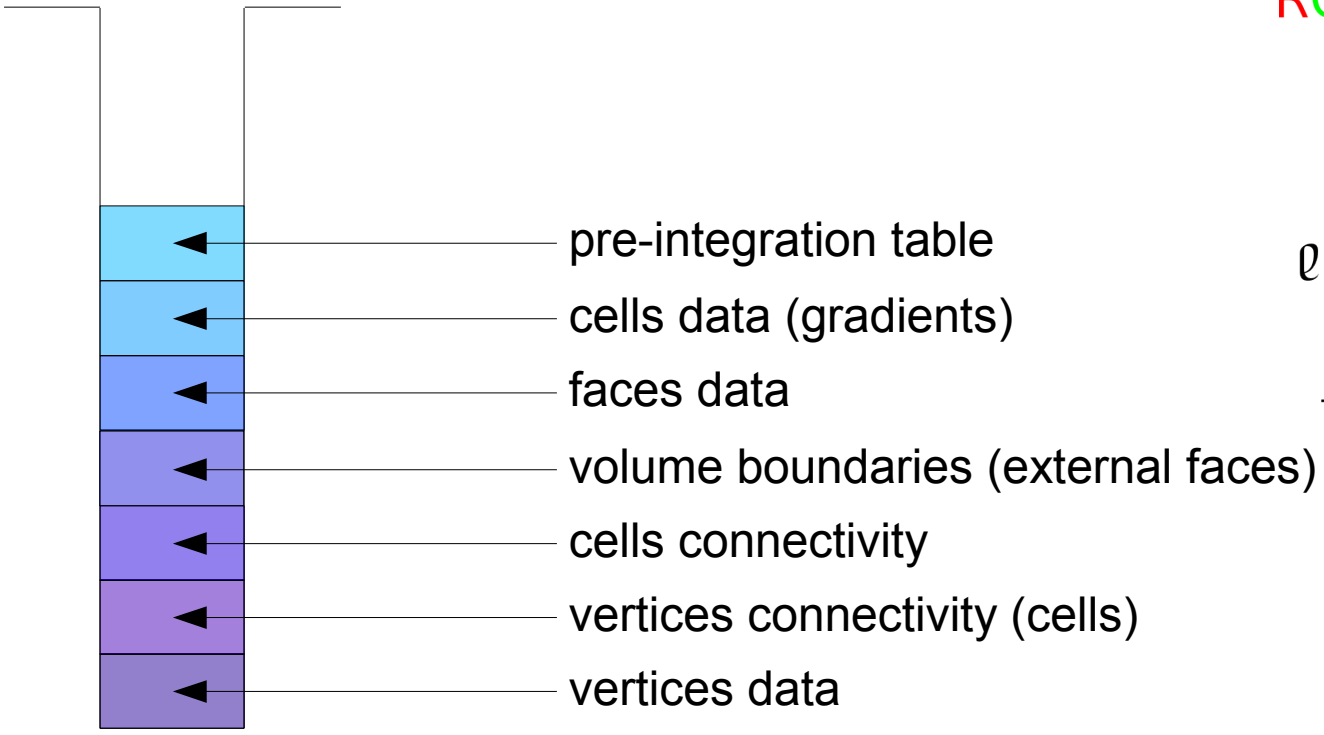


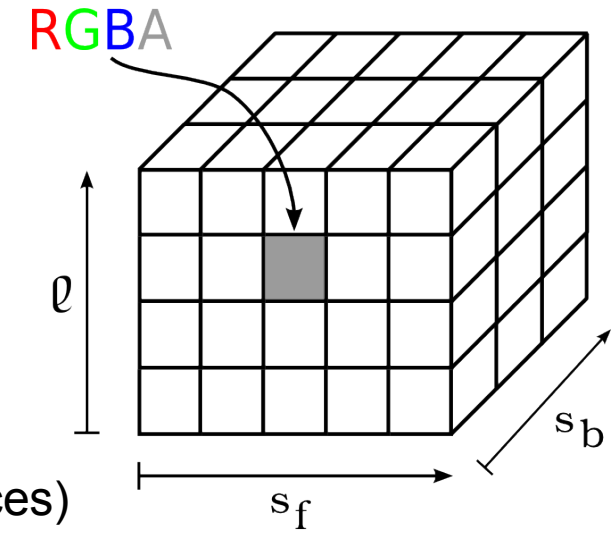


[ray integration]

Concerning Memory

- Volume Rendering “Memory Well”

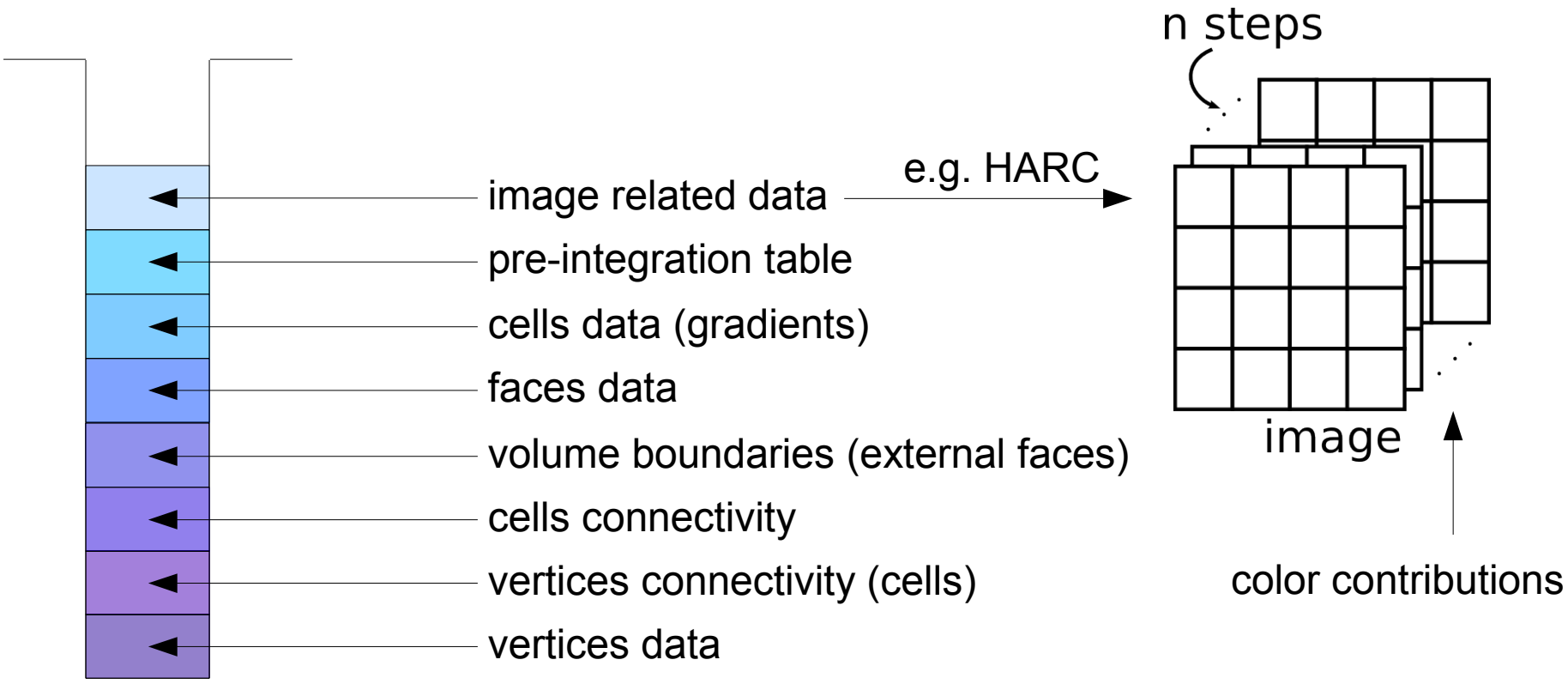




[ray integration]

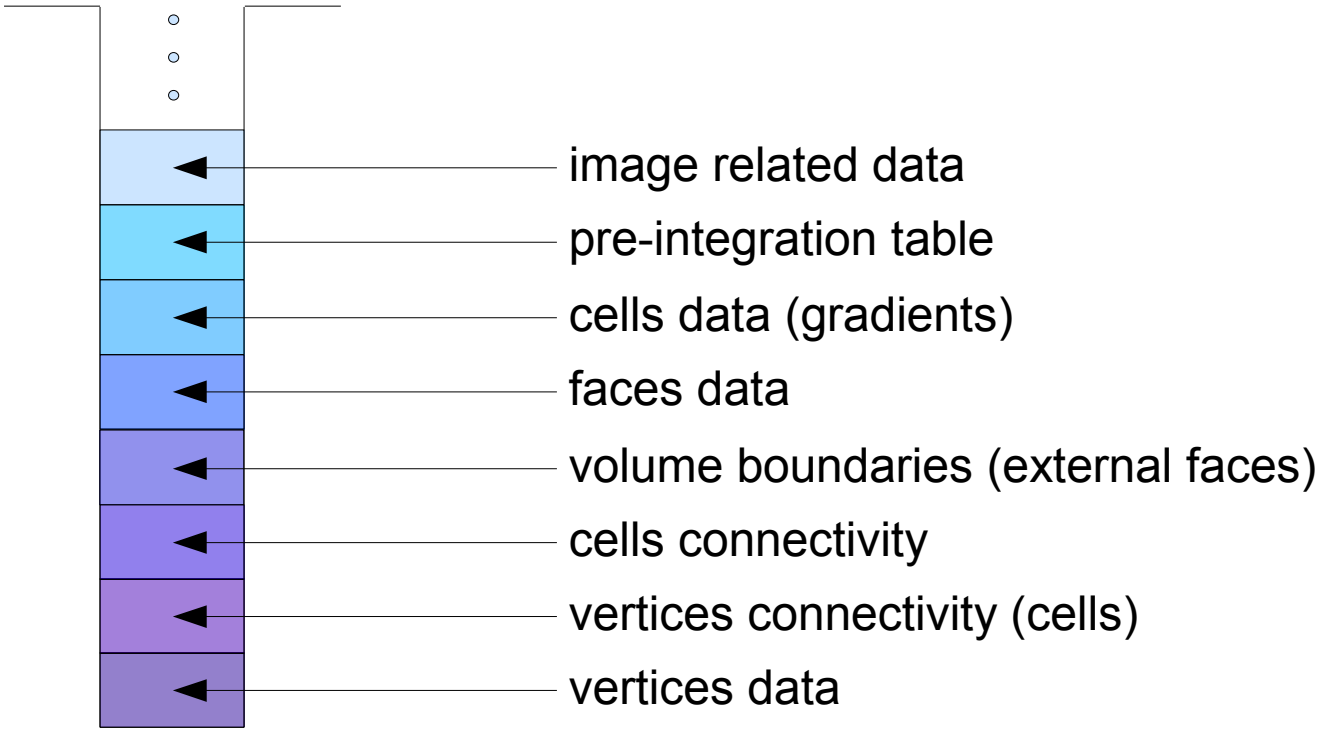
Concerning Memory

- Volume Rendering “Memory Well”



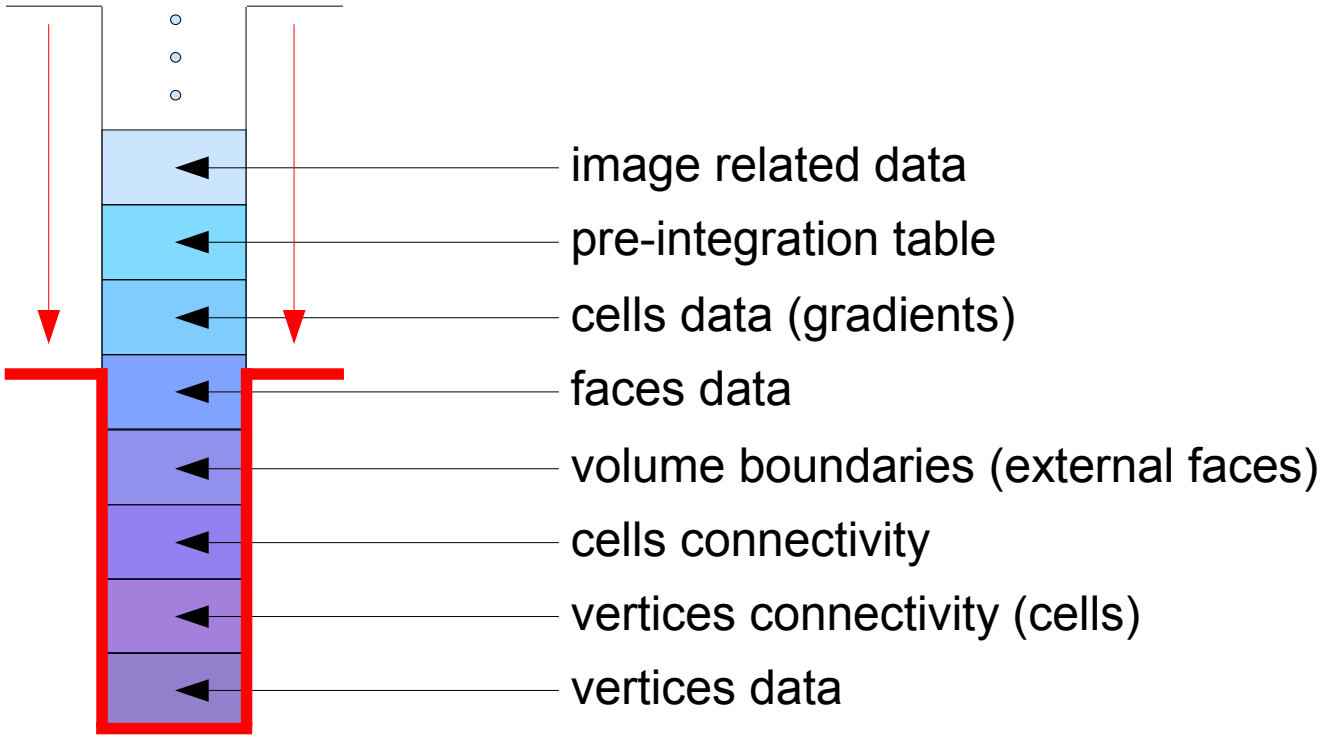
Concerning Memory

- Volume Rendering “Memory Well”



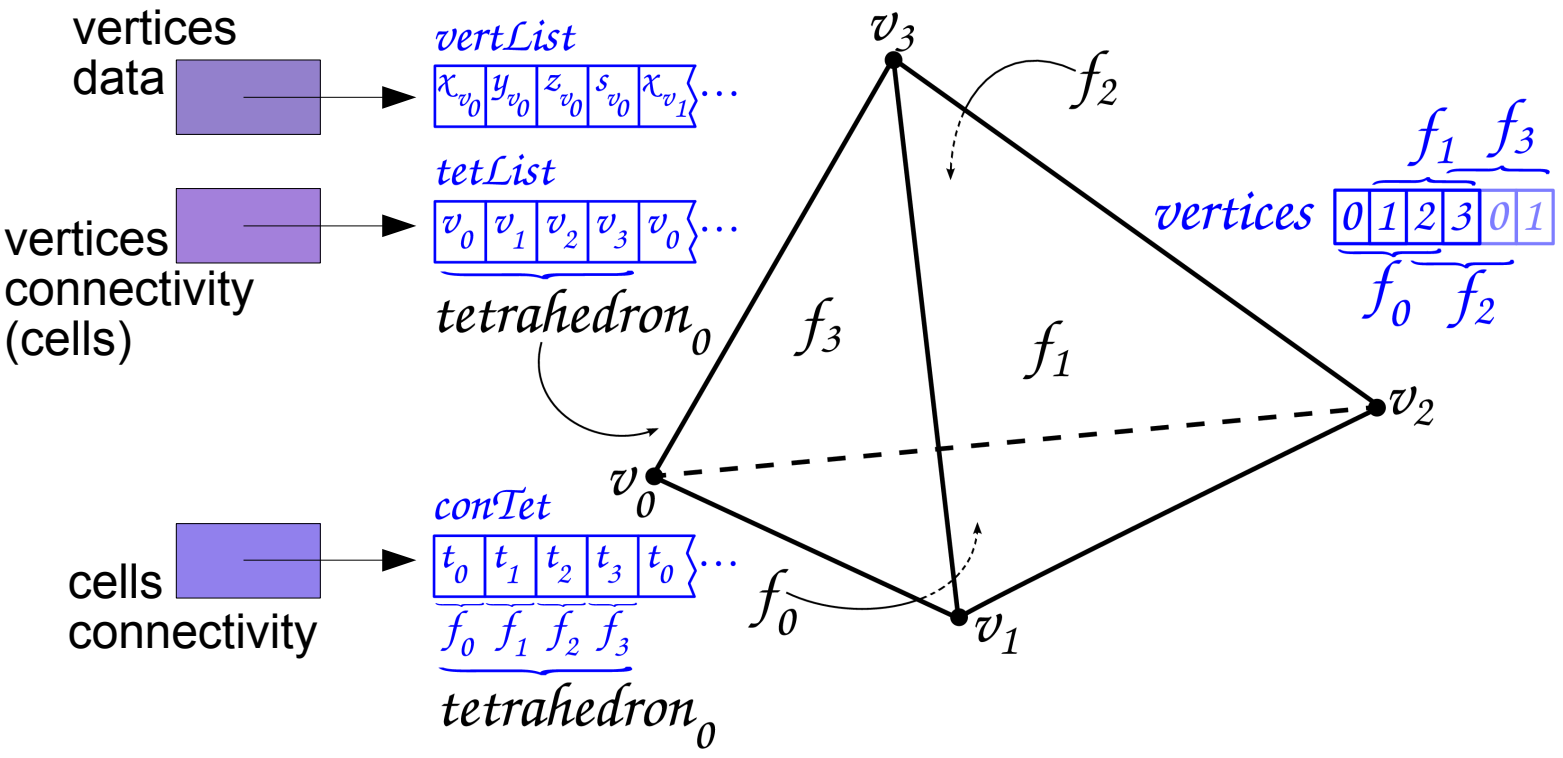
Concerning Memory

- Volume Rendering “Memory Well”



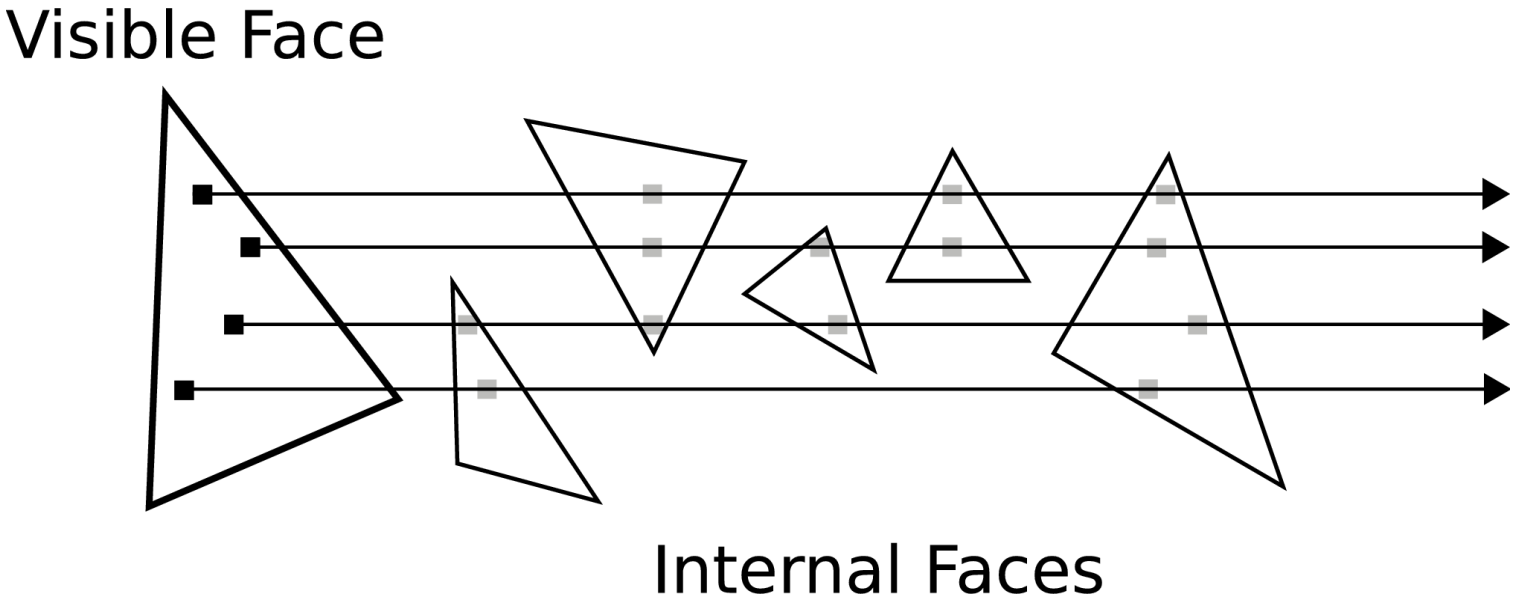
Basic Idea

- Data structures: a minimum approach



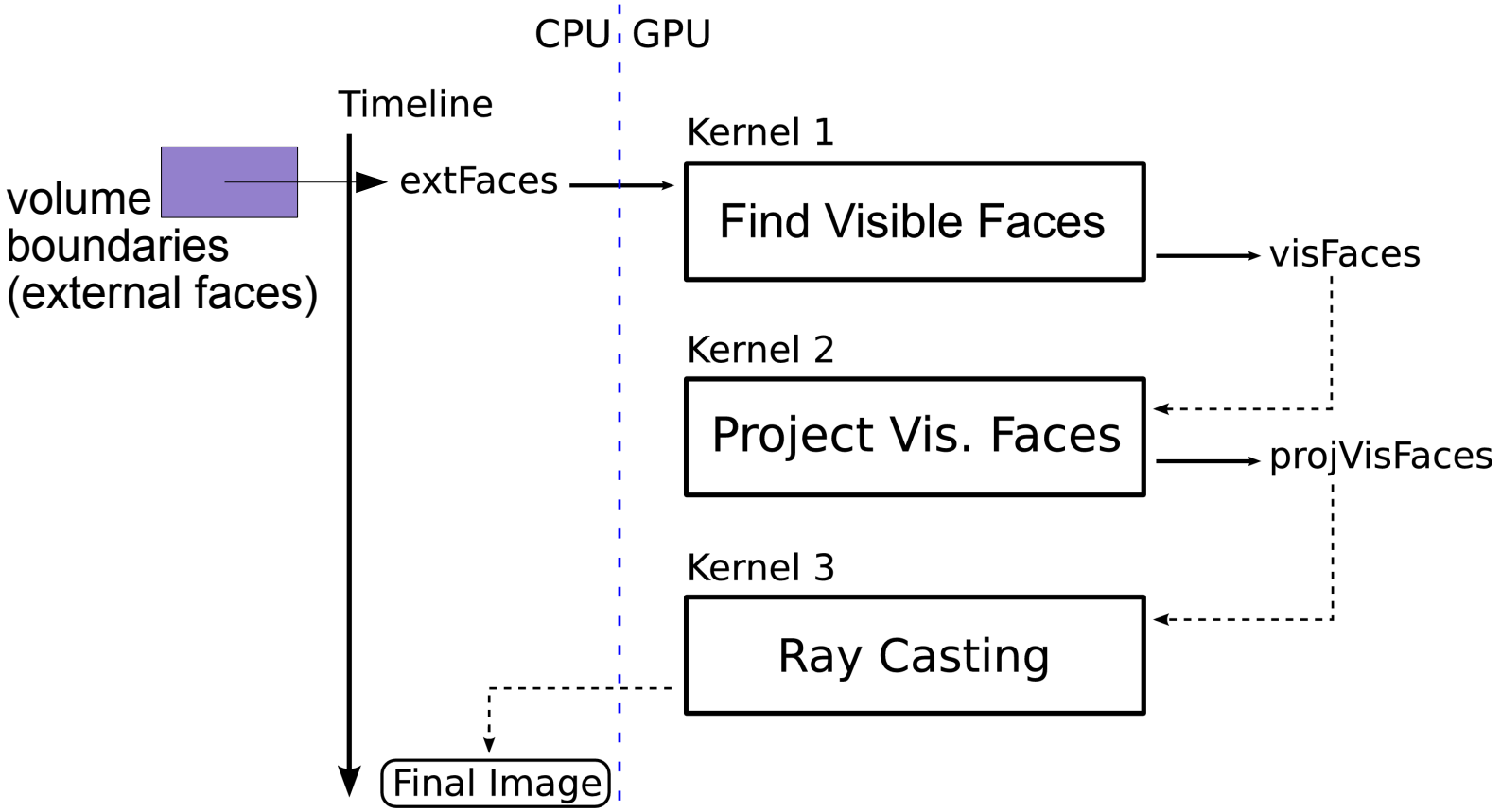
Ray Coherence – VF-Ray

- Building face data on-the-fly



Memory Efficient GPU-Based Ray Casting

- Algorithm Overview



Kernel 1/3 – Find Visible Faces

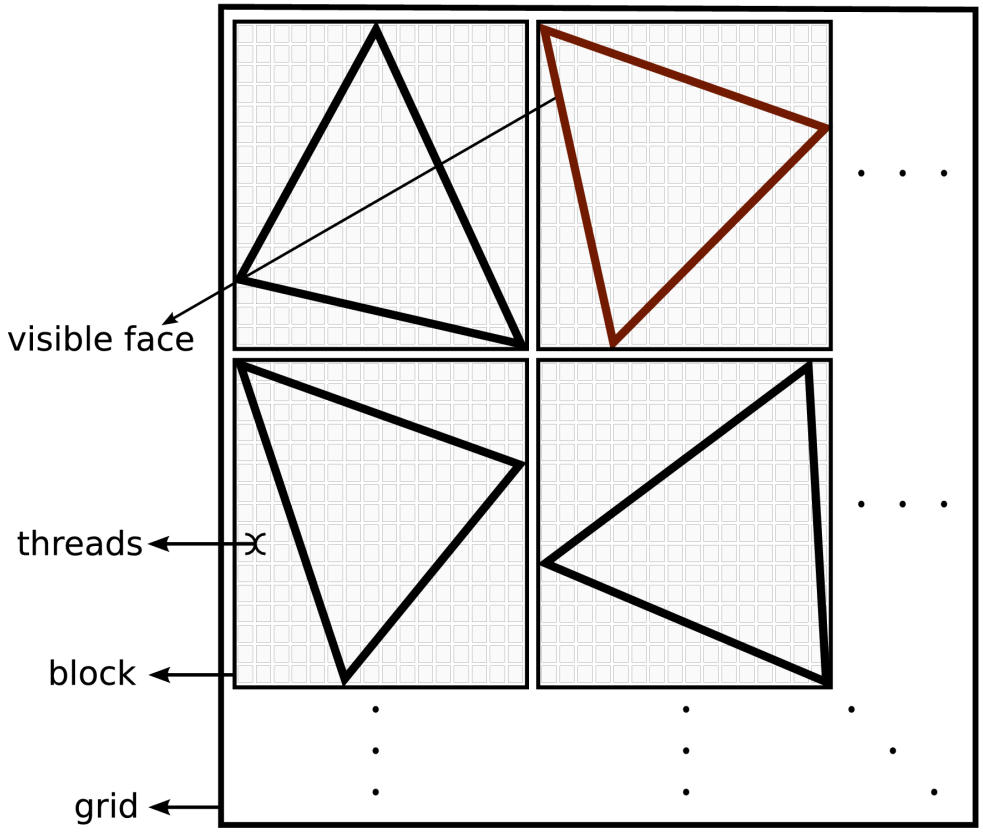
- Back face culling
- Thread – One external face
- Main role – Reduce the input of the next 2 kernels

Kernel 2/3 – Project Visible Faces

- Compute bounding box of pixels
- Thread – One visible face
- Main role – Avoid image related data

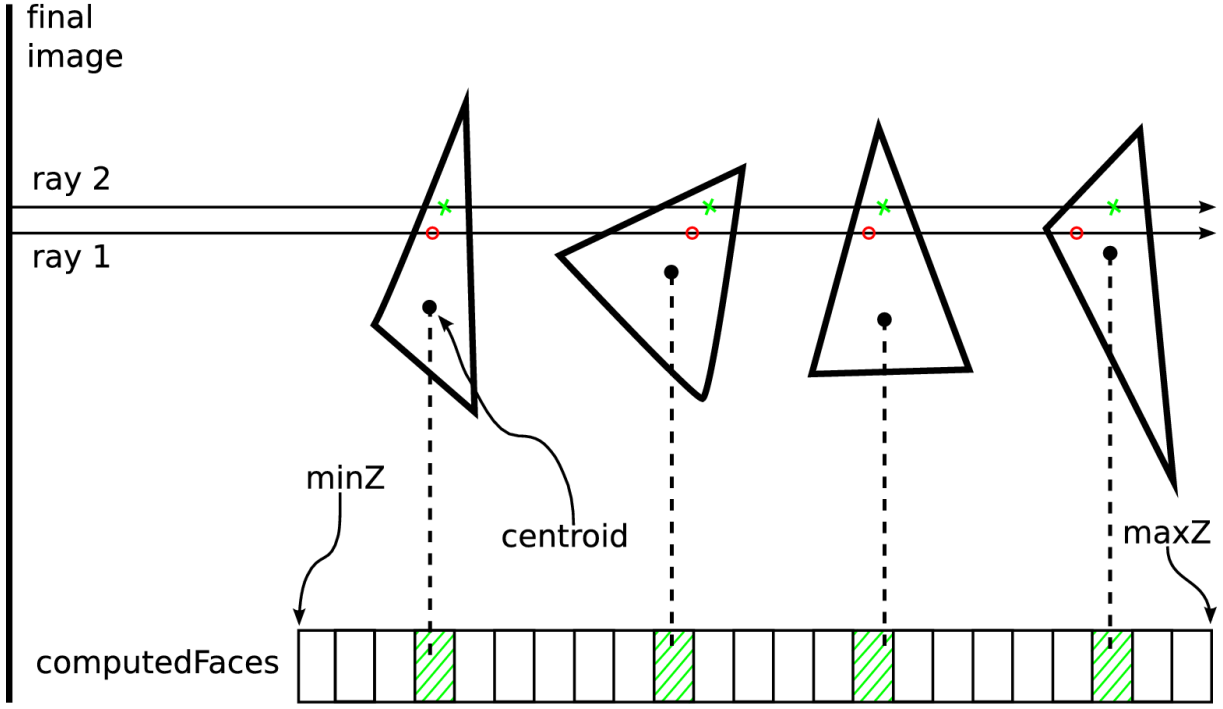
Kernel 3/3 – Ray Casting

- Compute ray casting algorithm
- Thread – Small bounding box of pixels
- Block – One visible face
- Main role – Render the final image



Kernel 3/3 – Buffer

- Ray coherence:
  - the *computedFaces* buffer



Results

- Hardware-based algorithms memory usage

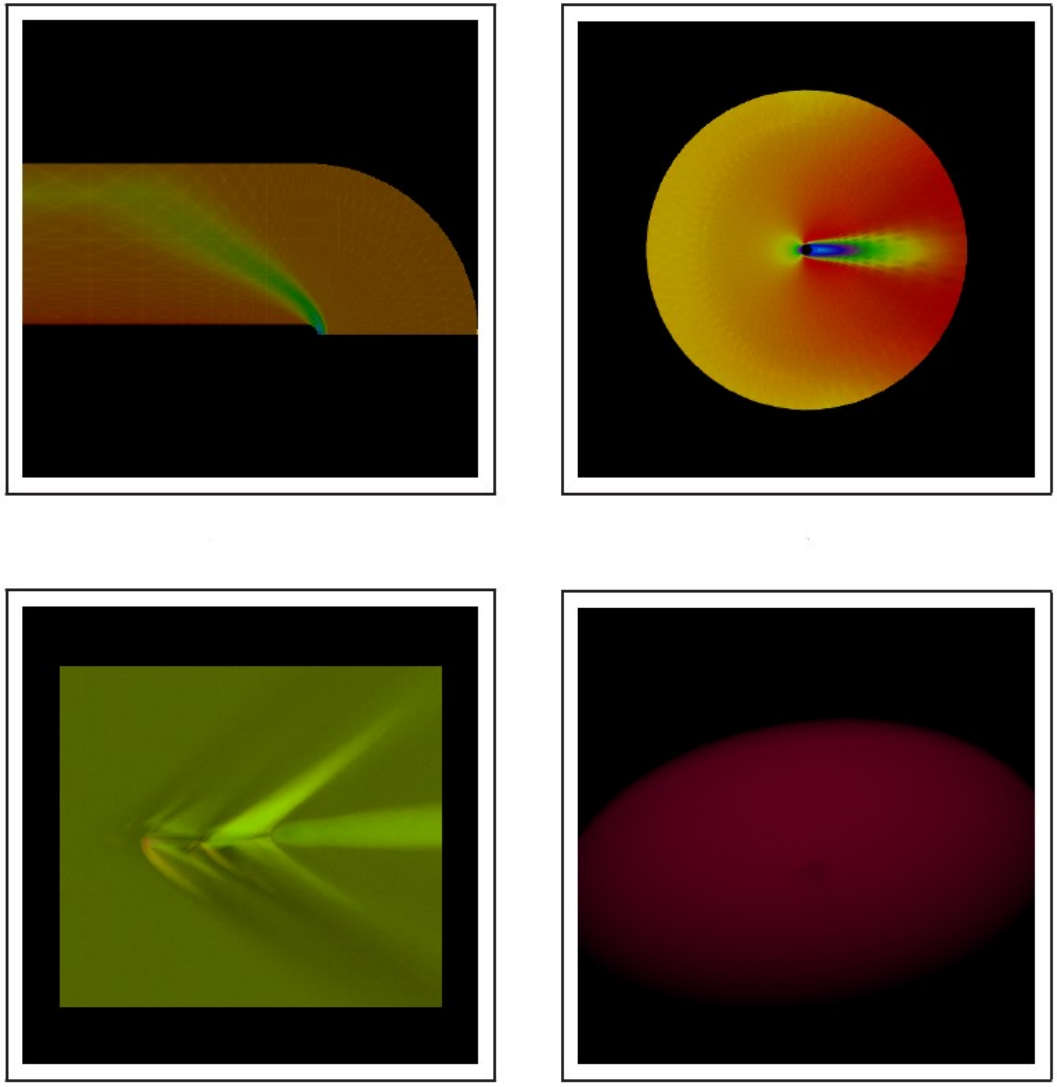
Algorithm	*Bytes/Tet*	*Bytes/Pixel*	*Pre-Int*
VICP	456	–	16
HARC	160	96	16
HARC-Partial	96	96	1
VF-Ray-GPU	38	–	–

Mesh related data

Image related data

Pre-integration table

Results



Results

- Tested datasets sizes

Dataset	# Verts	# Faces	# Boundary	# Cells
Blunt	41 K	381 K	13 K	187 K
Oxygen	109 K	1 M	27 K	513 K
F16	1.1 M	12.9 M	309 K	6.3 M
Fighter+	1.9 M	22.1 M	334 K	11.2 M

Results

- Memory and timing comparison

smaller datasets

Algorithm	Blunt Fin		Oxygen Post	
	Memory (KB)	*Time (ms)*	*Memory (KB)*	*Time (ms)*
VICP	118,524	190	249,928	546
HARC	72,267	18	123,245	33
HARC-Partial	22,636	32	50,248	51
VF-Ray-GPU	7,029	186	19,494	370

Results

- Memory and timing comparison

Datasets	Memory (MB)		Time (ms)	
	CPU	GPU	CPU	GPU
Fighter+	876	426	16263	3081
F16	499	239	2515	804

Conclusions

- Memory efficiency – 77% to 90% less memory
- Render very large datasets – over 10 Millions of cells *(nVidia GeForce 8800 Ultra)*
- Performance – faster than CPU version and GPU hybrid approach
- Comparison with HARC

Future Works

- Thread synchronization – Non-convex meshes
- Partial pre-integration – Improve quality
- Ways to improve performance

Memory Efficient GPU-Based Ray Casting for Unstructured Volume Rendering

Email: andmax@cos.ufrj.br

COPPE/UFRJ: http://www.cos.ufrj.br

LCG: http://www.lcg.ufrj.br

Thank you!

LABORATÓRIO DE COMPUTAÇÃO GRÁFICA
PESC / COPPE / UFRJ